AMENDMENTS TO THE SPECIFICATION

[0085] In this document what is meant by the terms ~~To be understood as~~ "computer" or "device"~~;~~ ~~in this document,~~ are, *inter alia*, all possible so-called Customer Premise Equipment (CPE): PC, routers, laptop, PDA, smart-phones etc.

[0086] In this document what is meant by the term ~~To be understood as~~ "socket" is~~, in this document, the~~ an Internet socket, i.e. one end of a bi-directional communication link between two programs defined as the combination of an IP address and a port number.

[0087] FIG. 5 to 11 illustrate an architecture that can be used to achieve the invention. The invention can be used by any Client application that is connected through an IP network (e.g. Internet or any wired/wireless Intranet) to its Server application (FIG. 5). The method and system according to the invention comprise one or more Client applications 11, running on a mobile device 10, that are connected via a local network provider 30/31/32/33 and through an IP network 34, e.g. the worldwide backbone network called Internet or any wired/wireless Intranet, to a Server application 21 (FIG. 6). The networks implemented by providers 30/31/32/33 can comprise Ethernet or another wired LAN (Local Area Network), Bluetooth, GSM (Global System for Mobile Communication), GPRS (Generalized Packet Radio Service), EDGE (Enhanced Data-Rates for GSM Evolution), 1XRTT (Radio Transmission Technology), CDMA2000 (Code Division Multiple Access), UMTS (Universal Mobile Telecommunications System) and/or WLAN (Wireless Local Area Network), etc. The term ~~To be understood as~~ "mobile device" 10 ~~are,~~ encompasses, *inter alia*, all possible devices (laptops, PDAs, smart-phones etc) and in general all so-called Customer Premise Equipment (CPE) equipped with one or more different physical network interfaces 40 to 43, which can also support the different network standards of the various network providers 30 to 33. The physical network interfaces 40 to 43 of the mobile device 10 can comprise e.g. interfaces for the mentioned networks as e.g. Ethernet or another wired LAN, Bluetooth, GSM, GPRS, EDGE, CDMA2000UMTS and/or WLAN, etc. The interfaces can be not only packet-switched interfaces, as used directly by network protocols such as e.g. Ethernet or Token Ring, but also circuit-switched interfaces which can be used by means of protocols such as e.g. PPP (Point-to-Point Protocol), SLIP (Serial Line Internet Protocol) or GPRS (Generalized

Packet Radio Service), i.e. whose interfaces do not have, for example, any network addresses such as a MAC or a DLC address. The invention provides any Client application with the best local network connection to the IP network 34 in term of bandwidth, reliability and cost effectiveness among all the wired/wireless network access technologies and/or access providers available at a certain time and location 30/31/32/33, managing the switch (when needed or convenient) in a transparent automatic and/or semi-automatic way without interrupting active network applications or sessions. In particular, the switch between the networks can comprise a change of a physical network interface among different networks technologies, such as e.g. Ethernet, Bluetooth, mobile radio networks (GSM, GPRS, EDGE, CDMA2000, UMTS, etc.) or WLAN, or also a topological location change within the same network technology, for example a device linked to an Ethernet network that migrate to another Ethernet network or a device linked to a Wi-Fi HotSpot that migrate to another Wi-Fi HotSpot. In particular the invention is suited for client network applications running on a mobile device 10 which is able to hold simultaneously at least two different kind of network connection (GSM, GPRS, EDGE, CDMA2000, UMTS, Satellite Links, Wi-Fi, LAN, PSTN [Public Switched Telephone Network], xDSL as ADSL [asymmetric digital subscriber line] or SDSL [symmetric digital subscriber line] etc): for instance, mobile devices with a GPRS modem and an independent Wi-Fi adapter or mobile devices with a Wi-Fi adapter and an independent UMTS modem, or so on. However, it is important to note that the invention also works with mobile devices able to hold only one network connection at a time (e.g. a laptop that possesses only one slot for insertion of a PCMCIA network card (PCMCIA: Personal Computer Memory Card International Association)[[)]]. According to the invention the client application and its server exchange data via an IP network, such as e.g. the Internet, using exactly one network connection at a time.

[0089] The invention avoids any interruption of service without any modification of the providers' infrastructure. One of the purposes of the invention is to make the IP address change of the mobile device totally transparent to the client and server applications. The invention acts as [[a]] middleware, making the client application ~~to~~ believe that it is running either on the same device as the server application or in a device directly connected to the server (depending on the configuration adopted). This is achieved through the client-service module 12, further called

CNAPT (Client Network Address and Port Translator) (FIG. 8), and a server-service module 22, further called SNAPT (Server Network Address and Port Translator) (FIG. 10). By means of the CNAPT module and the SNAPT module, if the provider X becomes unavailable (slowly or suddenly) while the network provider Y remains still available, and if there is enough time for the IP transition phase (that is to say the client and the server applications do not go into timeout), the switch between "ClientIP_X" and "ClientIP_Y" is totally transparent to the client and server applications that continue their execution without any session crash and/or interruption of service. Note that the IP transition phase can be generated also by a temporary interruption of the Internet connection provided by the network provider X, that however remains still available; in fact that temporary interruption may cause a modification on the assigned IP address, e.g. from "ClientIP_X" to "ClientIP_X1". During the IP transition phase: (1) The CNAPT stops forwarding all the outgoing IP packets generated by the Client application. They are buffered by the CNAPT and they will be forwarded at the end of the IP transition phase. (2) The SNAPT stops forwarding all the outgoing IP packets generated by the Server application and directed to the switching CNAPT. They are buffered by the SNAPT and they will be forwarded to the switching CNAPT at the end of the IP transition phase.

[0095] The CNAPT module not only acts as a layer 7 relay system. It also launches a "task" module in order to provide the client application with the best network connection, e.g. Internet connection, in term of bandwidth, reliability and cost effectiveness. The "task" module can be realized as a software and/or hardware module and has two main activities: (1) It continuously verifies the current network connection reliability and performance (Check Activity). (2) It continuously searches for new network providers (Search Activity). At any time, the user 1 can ask ~~to~~ the "decision task" module to switch to another available network provider. This is useful when the current network connection is a wired connection and the user 1 wants to switch to a wireless connection so that the user can unplugg~~ing~~ for instance the Ethernet cable. Note that the "decision task" can launch the Search and the Check activities only if it is running on a mobile device that is able to hold simultaneously at least two different kind of network connection (GSM, GPRS, EDGE, UMTS, Satellite Links, Wi-Fi, LAN, PSTN, ADSL . . . etc). If that mobile device is able to hold only one network connection at a time (e.g. a laptop that possesses only one slot for insertion of a PCMCIA network card), the Search activity can't be launched and the Check activity only verifies the reliability of the current Internet connection signalling to the user 1 its eventual unavailability. In this case, the switch from one network provider to another (or also, in case of temporary interruption, to the same network provider if it is became available in the meanwhile, e.g. there was only a temporary problem) can be decided only by the user 1 with a specific request to the "decision task".

[0109] In the following an embodiment of the invention based on the "Basic configuration" will be described in more detail. For this example of an embodiment, it is assumed that the mobile device 10 using the client application 11 is able to hold two different network connections simultaneously, e.g. Internet connections, through the network adapter 1 and 2 (40 and 41 in FIG. 17). The network adapter 1 and 2 are wireless adapters. In this example the server application 21 provides only two connection-oriented services (like services based on TCP): ServiceA and ServiceB (this assumption is used only to simplify the drawings; obviously the switch procedure also handles ~~also~~ the connectionless server services). For each of them a server socket 29 listens for client application requests (FIG. 17). The server device is connected to Internet 34 and it has the IP address "ServerIP". The mobile device 10, using the wireless adapter 1 [40], is connected to Internet 34 via the wireless provider 1 [30] and it has the IP address "ClientIP_1". The client application does not provide any client services (this assumption is used only to simplify the drawings; obviously the switch procedure handles also the connection-oriented/connectionless client services). The CNAPT module and the SNAPT module will be described as middleware software application installed respectively on the mobile device 10 and on the server PC 20. In a modified embodiment, the CNAPT module and the SNAPT module run on a dedicated device. In this case the client and the server applications send/receive packets to/from the external dedicated device. The unique difference from the other implementation is that the CNAPT module and SNAPT module are not referred to by the loopback address, but by their external dedicated device IP address.

[0118] When the SNAPT module receives a "SWITCH_STEP1" packet, it has to perform the following steps (FIG. 24):

- (SNAPT-1) Retrieve from this packet the ID of the sender CNAPT (from now on referred as "SWITCHING-CNAPT").
- (SNAPT-2) Stop forwarding the connection-oriented server service reply packets to the "SWITCHING-CNAPT".
- (SNAPT-3) Stop forwarding the connection-oriented client service request packets to the "SWITCHING-CNAPT".
- (SNAPT-4) Stop forwarding the connectionless client service packets to the "SWITCHING-CNAPT".
- (SNAPT-5) Buffer the pending server service replies and the pending client service requests. They will be forwarded at the end of the IP transition phase. (SNAPT-6) Flush all the transmission buffers.
- (SNAPT-7) Put the input stream of the connection-oriented sockets linked to the "SWITCHING-CNAPT" in "waiting for last packet before redirection" mode. These sockets continue to receive data from the "SWITCHING-CNAPT" as before until they receive a "LAST_MESSAGE_BEFORE_REDIRECTION" packet (note that this step can be avoided for connectionless services because they can tolerate packet losses).
- (SNAPT-8) Send a "LAST_MESSAGE_BEFORE_REDIRECTION" packet in each output stream of the connection-oriented sockets linked to the "SWITCHING-CNAPT.
- (SNAPT-9) Confirm to the "SWITCHING-CNAPT" control socket 122 the reception of the "SWITCH_STEP1" packet with an "ACK" packet.
- (SNAPT-10) Wait until all the input streams in "waiting for last packet before redirection" mode have received the "LAST_MESSAGE_BEFORE_REDIRECTION" packet.

[0119] When the CNAPT module receives the SNAPT module acknowledgement, it has to perform the following steps (FIG. 25):

- (CNAPT-12) Send a "LAST_MESSAGE_BEFORE_REDIRECTION" packet in each output stream of the connection-oriented sockets linked to the SNAPT.
- (CNAPT-13) Wait until all the input streams in "waiting for last packet before redirection" mode have received the "LAST_MESSAGE_BEFORE_REDIRECTION" packet.
- (CNAPT-14) Wait for SNAPT's "OK_TO_REDIRECTION" packet.

[0120] When the SNAPT module has received all the "LAST_MESSAGE_BEFORE_REDIRECTION" packets, it has to perform the following steps (FIG. 26):

- (SNAPT-11) Send an "OK_TO_REDIRECTION" packet to the "SWITCHING-CNAPT" control socket 122.
- (SNAPT-12) Wait for the "CLIENT_SERVICES_READY_FOR_REDIRECTION" packet coming from the "SWITCHING-CNAPT" control socket 122.

[0121] When the CNAPT module receives the "OK_TO_REDIRECTION" packet, it has to perform the following steps (FIG. 27):

- (CNAPT-15) Destroy the connectionless Client Services Emulator sockets 17 bound to the current CNAPT IP address and receiving data from the SNAPT.
- (CNAPT-16) While p[[P]]reserving the Client services connections between the Client and the Server applications, destroy the Client services connection-oriented emulation sockets linked to the SNAPT and generated by the connection-oriented Client Services Emulator server sockets 17. They were bound to the current CNAPT IP address. Each of these sockets has to be renewed by the SNAPT in order to preserve the Client services connections between the Client and the Server applications.
- (CNAPT-17) While p[[P]]reserving the Client services connections between the Client and the Server applications, destroy the connection-oriented Client Services Emulator server sockets 17 accepting requests from the SNAPT. They were bound to the current CNAPT IP address.

- (CNAPT-18) Send a "CLIENT_SERVICES_READY_FOR_REDIRECTION" packet to the SNAPT's control server socket 27.

- (CNAPT-19) Wait for the "SERVER_SERVICES_READY_FOR_REDIRECTION" packet coming from the SNAPT's control server socket 27.

[0122] When the SNAPT module receives the "CLIENT_SERVICES_READY_FOR_REDIRECTION" packet, it has to perform the following steps (FIG. 28):

- (SNAPT-13) While p[[P]]reserving the Client services connections between the Client and the Server applications, destroy the connection-oriented Client Services Server Request Emulation sockets 61 linked to the current "SWITCHING-CNAPT" IP address.

- (SNAPT-14) While p[[P]]reserving the Server services connections between the Client and the Server applications, destroy the Server services connection-oriented emulation sockets linked to the current "SWITCHING-CNAPT" IP address and generated by the connection-oriented Server Services Emulator server sockets 26. Each of these sockets has to be renewed by the "SWITCHING-CNAPT" in order to preserve the Server services connections between the Client and the Server applications.

- (SNAPT-15) Send a "SERVER_SERVICES_READY_FOR_REDIRECTION" packet to the "SWITCHING-CNAPT" control socket 122.

- (SNAPT-16) Wait for the "SWITCH_STEP2" packet coming from the "SWITCHING-CNAPT" through the new control socket 122 bound to its new IP address.

[0123] When the CNAPT module receives the "SERVER_SERVICES_READY_FOR_REDIRECTION" packet, it has to perform the following steps (FIG. 29):

- (CNAPT-20) While p[[P]]reserving the Server services connections between the Client and the Server applications, destroy the connection-oriented Server services Client Request Emulation sockets 121 bound to the current "SWITCHING-CNAPT" IP address.

- (CNAPT-21) Destroy the control socket 122 bound to the current "SWITCHING-CNAPT" IP address. (CNAPT-22) If needed, close the old Internet connection through the Wireless Provider 1 (if the client application and the server application are connected through a VPN, this step has to be preceded by the following step: Close the existing VPN connection between "ClientIP_1" and "ServerIP").

- (CNAPT-23) If the "ClientIP_2" address is not yet available, open the new Internet connection through the selected wireless adapter (in this case the Wireless Adapter 2). (CNAPT-24) Change the current IP address from the old IP address to the new one (if the client application and the server application are connected through a VPN, this step has to be followed by the following step: Open the new VPN between "ClientIP_2" and "ServerIP").

- (CNAPT-25) Change to the new CNAPT IP address the binding of the server services connectionless sockets transmitting to the SNAPT module.

- (CNAPT-26) Recreate the connectionless Client Services Emulator sockets 17 receiving data from the SNAPT and bind them to the new CNAPT IP address.

- (CNAPT-27) Preserving the Client services connections between the Client and the Server applications, recreate the connection-oriented Client Services Emulator server sockets 17 accepting requests from the SNAPT and bind them to the new CNAPT IP address.

- (CNAPT-28) Create a new control socket 122 "ClientIP_2:CasualPort" and connect it to the control server socket 27 provided by SNAPT module on "ServerIP:ControlPort".

- (CNAPT-29) Send a "SWITCH_STEP2" packet to the SNAPT's control server socket 27 using the Wireless Adapter 2. This packet contains the CNAPT ID and through this packet the SNAPT module can deduce the new CNAPT IP address.

- (CNAPT-30) Wait for the renewal of the connection-oriented Client services emulation sockets destroyed in [CNAPT-16]. This renewal will be done with a connection request [SNAPT-19] of the SNAPT for each connection-oriented socket generated by the Client Services Emulator server sockets 17.

[0124] When the SNAPT module receives the "SWITCH_STEP2" packet, it has to perform the following steps:

- (SNAPT-17) Update the "SWITCHING-CNAPT" IP address with its new IP address as retrieved by the source field of the "SWITCH_STEP2" control packet.
- (SNAPT-18) Redirect the Client services connectionless sockets transmitting to the "SWITCHING-CNAPT" from its old IP address to the new one (no redirection is needed for the connectionless server sockets receiving data from the "SWITCHING-CNAPT"; they are unaffected by the CNAPT IP address switch).
- (SNAPT-19) Recreate the connection-oriented Client Services Server Request Emulation sockets 61 destroyed in [SNAPT-13] and connect them, in order to preserve the Client/Server interaction, to the Client Services Emulator server sockets 17 provided on the new "SWITCHING-CNAPT" IP address.
- (SNAPT-20) Wait for the renewal of the connection-oriented Server services emulation sockets destroyed in [SNAPT-14]. This renewal will be done with a connection request [CNAPT-32] of the "SWITCHING_CNAPT" for each connection-oriented socket generated by the Server Services Emulator server sockets 26.

[0125] When the CNAPT module has received the renewal of the connection-oriented Client services emulation sockets destroyed in [CNAPT-16] it has to perform the following steps (FIG. 30):

- (CNAPT-31) Redirect the correspondent Client services emulation connections to the new IP address through the renewed emulation sockets in order to preserve the Client/Server interaction.
- (CNAPT-32) Recreate the connection-oriented Server services Client Request Emulation sockets 121 destroyed in [CNAPT-20], bind them to the new "SWITCHING-CNAPT" IP address and connect them, in order to preserve the Client/Server interaction, to the Server Services Emulator server sockets 26 provided on the SNAPT IP address.

- (CNAPT-33) Wait for the "ALL_REDIRECTED" packet coming from the SNAPT's control server socket 27.

[0126] When the SNAPT module receives the renewal of the connection-oriented Server services emulation sockets destroyed in [SNAPT-14] it has to perform the following steps (FIG. 31):

- (SNAPT-21) Redirect the correspondent Server services emulation connections to the new "SWITCHING-CNAPT" IP address through the renewed emulation sockets in order to preserve the Client/Server interaction.
- (SNAPT-22) Send an "ALL_REDIRECTED" packet to the new "SWITCHING-CNAPT" control socket 122.
- (SNAPT-23) Wake up all the connectionless/connection-oriented client and server services and send the packets buffered in [SNAPT-5].

[0127] When the CNAPT module receives the "ALL_REDIRECTED" packet, it has to perform the following steps (FIG. 32):

- (CNAPT-34) Wake up all the connectionless/connection-oriented client and server services and the "decision task" activities (Search and Check activities) and send the packets buffered in [CNAPT-5].

[0132] When the CNAPT catches the connection exception or when the Check Activity signals the interruption, the CNAPT has to establish a new Internet connection through the Wireless Provider 2 (or, in turn, through the Wireless Provider 1 if it is became available in the meanwhile, e.g. there was only a temporary problem). When the new Internet connection is available, to begin the IP transition phase from the "ClientIP_1" (no more available) to the "ClientIP_2" the CNAPT has to perform the following steps:

- (CNAPT-1*) Suspend the "decision task" activities (Search and Check activities).
- (CNAPT-2*) Stop forwarding the connection-oriented Server services request packets to the SNAPT.

- (CNAPT-3*) Stop forwarding the connection-oriented Client services reply packets to the SNAPT.
- (CNAPT-4*) Stop forwarding the connectionless Server services packets to the SNAPT.
- (CNAPT-5*) Buffer the pending Server service requests and the pending Client service replies. They will be forwarded at the end of the IP transition phase.
- (CNAPT-6*) For each outgoing connection, store the eventual unsent packets (i.e. the packets that the outgoing connection was eventually sending when the interruption exception was caught).
- (CNAPT-7*) Wait until the Search/Check activities have been suspended.
- (CNAPT-8*) Destroy the connectionless Client Services Emulator sockets 17 bound to the old CNAPT IP address.
- (CNAPT-9*) While p[[P]]reserving the Client services connections between the Client and the Server applications, destroy the Client services connection-oriented emulation sockets linked to the SNAPT and generated by the connection-oriented Client Services Emulator server sockets 17. They were bound to the old CNAPT IP address. Each of these sockets has to be renewed by the SNAPT in order to preserve the Client services connections between the Client and the Server applications.
- (CNAPT-10*) While p[[P]]reserving the Client services connections between the Client and the Server applications, destroy the connection-oriented Client Services Emulator server sockets 17 accepting requests from the SNAPT. They were bound to the old CNAPT IP address.
- (CNAPT-11*) While p[[P]]reserving the Server services connections between the Client and the Server applications, destroy the connection-oriented Server services Client Request Emulation sockets 121 bound to the old CNAPT IP address.
- (CNAPT-12*) Change the current IP address from the old IP address to the new one (if the client application and the server application are connected through a VPN,

this step has to be followed by the following step: Open the new VPN between "ClientIP_2" and "ServerIP").

- (CNAPT-13*) Change to the new CNAPT IP address the binding of the Server services connectionless sockets transmitting to the SNAPT.
- (CNAPT-14*) Recreate the connectionless Client Services Emulator sockets 17 receiving data from the SNAPT and bind them to the new CNAPT IP address.
- (CNAPT-15*) Recreate the connection-oriented Client Services Emulator server sockets 17 accepting requests from the SNAPT and bind them to the new CNAPT IP address.
- (CNAPT-16*) Create a new Control socket "ClientIP_2:CasualPortK" and connect it to the Control server socket provided by SNAPT on "ServerIP:ControlPort".
- (CNAPT-17*) Send a "SWITCH_UNEXPECTED" packet to the SNAPT "Control server socket" using the new Internet connection. This packet contains the CNAPT ID and through this packet the SNAPT can deduce the new CNAPT IP address.
- (CNAPT-18*) Wait for the renewal of the connection-oriented Client services emulation sockets destroyed in [CNAPT-9*]. This renewal will be done with a connection request [SNAPT-11*] of the SNAPT for each connection-oriented socket generated by the Client Services Emulator server sockets 17.

[0133] When the SNAPT catches the connection exception it waits for a "SWITCH_UNEXPECTED" packet coming from that CNAPT ID. The SNAPT associates to that CNAPT ID a max reconnection waiting time (switch timeout). When this reconnection waiting time has elapsed, the related CNAPT will be considered definitely disconnected and the SNAPT should execute these steps: (1) release the internal resources of the disconnected CNAPT ID and remove it from the list of connected CNAPT; (2) destroy the Client service emulator server sockets related to the CNAPT ID and unbind them from the allocated Virtual IP address. When, in any way, the SNAPT receives a "SWITCH_UNEXPECTED" packet it has to perform the following steps:

- (SNAPT-1*) Retrieve from this packet the ID of the sender CNAPT (from now on referred as "SWITCHING-CNAPT"). Through this packet the SNAPT can also deduce the new "SWITCHING-CNAPT" IP address.

- (SNAPT-2*) Stop forwarding the connection-oriented Server service reply packets to the "SWITCHING-CNAPT".

- (SNAPT-3*) Stop forwarding the connection-oriented Client service request packets to the "SWITCHING-CNAPT".

- (SNAPT-4*) Stop forwarding the connectionless Client service packets to the "SWITCHING-CNAPT".

- (SNAPT-5*) Buffer the pending Server service replies and the pending Client service requests. They will be forwarded at the end of the IP transition phase.

- (SNAPT-6*) For each outgoing connection, store the eventual unsent packets (i.e. the packets that the outgoing connection was sending when the interruption exception was caught).

- (SNAPT-7*) While p[[P]]reserving the Client services connections between the Client and the Server applications, destroy the connection-oriented Client Services Server Request Emulation sockets 61 linked to the old "SWITCHING-CNAPT" IP address.

- (SNAPT-8*) While p[[P]]reserving the Server services connections between the Client and the Server applications, destroy the Server services connection-oriented emulation sockets linked to the old "SWITCHING-CNAPT" IP address and generated by the connection-oriented Server Services Emulator server sockets 26. Each of these sockets has to be renewed by the "SWITCHING-CNAPT" in order to preserve the Server services connections between the Client and the Server applications.

- (SNAPT-9*) Update the "SWITCHING-CNAPT" IP address with its new IP address as retrieved by the source address field of the "SWITCH_UNEXPECTED" control packet.

- (SNAPT-10*) Redirect the Client services connectionless sockets transmitting to the "SWITCHING-CNAPT" from its old IP address to the new one (No redirection is needed for the connectionless Server services sockets receiving data from the "SWITCHING-CNAPT". They are unaffected by the CNAPT IP address switch.).

- (SNAPT-11*) Recreate the connection-oriented Client Services Server Request Emulation sockets 61 destroyed in [SNAPT-7*] and connect them, in order to preserve the Client/Server interaction, to the Client Services Emulator server sockets 17 provided on the new "SWITCHING-CNAPT" IP address.
- (SNAPT-12*) Wait for the renewal of the connection-oriented Server services emulation sockets destroyed in [SNAPT-8*]. This renewal will be done with a connection request [CNAPT-21*] of the "SWITCHING_CNAPT" for each connection-oriented socket generated by the Server Services Emulator server sockets 26.

[0134] When the CNAPT has received the renewal of the connection-oriented Client services emulation sockets destroyed in [CNAPT-9*] it has to perform the following steps:

- (CNAPT-19*) Redirect the correspondent Client services emulation connections to the new IP address through the renewed emulation sockets in order to preserve the Client/Server interaction.
- (CNAPT-20*) Recreate the connection-oriented Server services Client Request Emulation sockets 121 destroyed in [CNAPT-11*], bind them to the new "SWITCHING-CNAPT" IP address and connect them, in order to preserve the Client/Server interaction, to the Server Services Emulator server sockets 26 provided on the SNAPT IP address.
- (CNAPT-21 *) Wait for the "ALL_REDIRECTED" packet coming from the SNAPT "Control server socket".

[0135] When the SNAPT receives the renewal of the connection-oriented Server services emulation sockets destroyed in [SNAPT-8*] it has to perform the following steps:

- (SNAPT-13*) Redirect the correspondent Server services emulation connections to the new "SWITCHING-CNAPT" IP address through the renewed emulation sockets in order to preserve the Client/Server interaction.
- (SNAPT-14*) Send an "ALL_REDIRECTED" packet to the new "SWITCHING-CNAPT" control socket.
- (SNAPT-15*) Resend the unsent packets eventually stored in [SNAPT-6*].
  (SNAPT-16*) Wake up all the connectionless/connection-oriented Client and Server services and send the packets buffered in [SNAPT-5*].

[0136] When the CNAPT receives the "ALL_REDIRECTED" packet it has to perform the following steps:

- (CNAPT-22*) Resend the unsent packets eventually stored in [CNAPT-6*].
- (CNAPT-23*) Wake up all the connectionless/connection-oriented Client and Server services and the "decision task" activities (Search and Check activities) and send the packets buffered in [CNAPT-5*].